IN THE CLAIMS:

Please amend the claims as follows:

1. (Currently Amended) A computer-implemented method for managing memory available for dynamic allocation during execution of code containing a plurality of memory allocators and a plurality of memory deallocators, comprising:

providing a data structure configured to record relationships between the memory deallocators and the memory allocators, wherein each of the relationships specify a memory allocator and a memory deallocator, and wherein each of the relationships requires at least one of: (i) that memory space allocated by the specified memory allocator is freed by the specified memory deallocator; and (ii) that memory space freed by the specified memory deallocator was allocated by the specified memory allocator; and wherein the data structure stores, for each relationship, a reference to both the specified memory deallocator and the specified memory allocator of the respective relationship;

providing a computer user interface;

allowing a user to establish, via the computer user interface, a relationship between ~~one or more of the~~ a memory ~~deallocators~~ deallocator and ~~one or more of the~~ a memory ~~allocators~~ allocator, the relationship being stored in the provided data structure~~, wherein the relationship requires that memory space allocated by the one or more allocators is freed by the one or more deallocators and wherein the relationship is represented by a data structure containing a reference to the one or more of the memory deallocators and the one or more of the memory allocators~~;

allowing the code to execute;

upon a call to ~~the one or more deallocators~~ a deallocator to free a memory space, determining, by operation of one or more computer processors, whether the call violates the user-established relationship ~~is violated~~, wherein determining whether the call violates the user-established relationship ~~is violated~~ comprises:

determining an associated allocator responsible for allocating the memory space; and

determining ~~that~~ whether the ~~memory space was allocated by an allocator different from the one or more memory allocators~~ user-established relationship specifies only one of the associated allocator and the called deallocator; and

upon determining that the call violates the user-established relationship ~~if so~~, notifying the user.

2. (Original) The method of claim 1, wherein notifying the user comprises halting execution of the code.

3. (Original) The method of claim 1, wherein notifying the user comprises halting execution of the code and displaying a status message to the user.

4. (Currently Amended) The method of claim 1, if the user-established relationship is not violated, freeing the memory space.

5. (Cancelled)

6. (Currently Amended) A computer-implemented method for managing memory available for dynamic allocation during execution of code containing a plurality of memory allocators and a plurality of memory deallocators, comprising:

providing a data structure configured to record relationships between the memory deallocators and the memory allocators, wherein each of the relationships specify a memory allocator and a memory deallocator, and wherein each of the relationships requires at least one of: (i) that memory space allocated by the specified memory allocator is freed by the specified memory deallocator; and (ii) that memory space freed by the specified memory deallocator was allocated by the specified memory allocator; and wherein the data structure stores, for each relationship, a reference to both the

specified memory deallocator and the specified memory allocator of the respective relationship;

establishing a relationship between a user-selected memory deallocator and a user-selected memory allocator, the relationship being stored in the provided data structure, ~~wherein the relationship requires that memory space freed by the user-selected deallocator have been allocated by the user-selected allocator, and wherein the relationship is represented by a data structure containing a reference to the user-selected deallocator and the user-selected allocator, and wherein the relationship is violated upon determining that the memory space freed by the user-selected deallocator was allocated by an allocator different from the user-selected allocator~~;

allowing the code to execute;

upon a call to the user-selected memory deallocator to free a memory space, determining, by operation of one or more computer processors, whether the memory space was allocated by the user-selected memory allocator; and

if not, notifying the user that the relationship is violated.

7. (Original) The method of claim 6, wherein notifying the user comprises halting execution of the code and displaying a status message to the user.

8. (Currently Amended) A method for managing memory available for dynamic allocation during execution of code containing a plurality of memory allocators and a plurality of memory deallocators, comprising:

providing a data structure configured to record upper limits on the amount of memory space that the memory allocators can allocate during execution of the code, each upper limit being specific to one of the memory allocators, wherein the data structure stores, for each memory allocator, both: (i) a reference to the respective memory allocator and (ii) the upper limit for the respective memory allocator;

setting an upper limit for ~~on the amount of memory space an~~ a memory allocator ~~can allocate during execution of the code, wherein the upper limit is specific to the allocator;~~, wherein both the upper limit and a reference to the memory allocator are

stored in [[a]] the provided data structure~~, thereby relating the upper limit to the allocator~~;

during execution of the code, tracking, by operation of one or more computer processors, the amount of memory space allocated by the memory allocator; and

when the amount of memory space allocated exceeds the upper limit, notifying a user.

9. (Currently Amended) The method of claim 8, wherein the step of tracking comprises: determining whether the memory allocator is called to allocate memory and, if so, incrementing a counter; and

determining whether a memory deallocator is called to deallocate memory allocated by the memory allocator and, if so, decrementing the counter.

10. (Currently Amended) The method of claim 8, wherein the step of tracking comprises incrementing a counter in the event of memory allocation by the memory allocator and decrementing the counter in the event of memory deallocation of memory space allocated by the memory allocator.

11. (Original) The method of claim 8, wherein notifying the user comprises halting execution of the code.

12. (Original) The method of claim 8, wherein the upper limit is independent of other memory size limitations.

13. (Original) The method of claim 8, wherein the upper limit is not a limit on a stack size.

14-22. (Canceled)

23. (Currently Amended) A computer readable storage medium containing a program which, when executed, performs an operation for managing memory available for dynamic allocation during execution of code containing a plurality of memory allocators and a plurality of memory deallocators, the operation comprising:

providing a data structure configured to record relationships between the memory deallocators and the memory allocators, wherein each of the relationships specify a memory allocator and a memory deallocator, and wherein each of the relationships requires at least one of: (i) that memory space allocated by the specified memory allocator is freed by the specified memory deallocator; and (ii) that memory space freed by the specified memory deallocator was allocated by the specified memory allocator; and wherein the data structure stores, for each relationship, a reference to both the specified memory deallocator and the specified memory allocator of the respective relationship;

establishing a relationship between a user-selected memory deallocator and a user-selected memory allocator, the relationship being stored in the provided data structure ~~wherein the relationship is represented by a data structure containing a reference to the user-selected deallocator and the user-selected allocator, and wherein the relationship requires that memory space freed by the user-selected deallocator have been allocated by the user-selected allocator~~;

allowing the code to execute;

upon a call to the user-selected memory deallocator to free a memory space, determining, by operation of one or more computer processors, whether the memory space was allocated by the user-selected memory allocator; and

if ~~so~~ not, notifying the user that the relationship is violated.

24. (Previously Presented) The computer readable storage medium of claim 23, wherein notifying the user comprises halting execution of the code and displaying a status message to the user.

25. (Currently Amended) A computer readable storage medium containing a program which, when executed, performs an operation for managing memory available for dynamic allocation during execution of code containing a plurality of memory allocators and a plurality of memory deallocators, the operation comprising:

providing a data structure configured to record upper limits on the amount of memory space that the memory allocators can allocate during execution of the code, each upper limit being specific to one of the memory allocators, wherein the data structure stores, for each memory allocator, both: (i) a reference to the respective memory allocator and (ii) the upper limit for the respective memory allocator;

setting an upper limit for ~~on the amount of memory space an~~ a memory allocator ~~can allocate during execution of the code, wherein the upper limit is specific to the allocator~~, wherein both the upper limit and a reference to the memory allocator are stored in the provided data structure;

during execution of the code, tracking, by operation of one or more computer processors, the amount of memory space allocated by the memory allocator; and

when the amount of memory space allocated exceeds the upper limit, notifying a user.

26. (Currently Amended) The computer readable storage medium of claim 25, wherein the step of tracking comprises:

determining whether the memory allocator is called to allocate memory and, if so, incrementing a counter; and

determining whether a memory deallocator is called to deallocate memory allocated by the memory allocator and, if so, decrementing the counter.

27. (Currently Amended) The computer readable storage medium of claim 25, wherein the step of tracking comprises incrementing a counter in the event of memory allocation by the memory allocator and decrementing the counter in the event of memory deallocation of memory space allocated by the memory allocator.

28. (Previously Presented) The computer readable storage medium of claim 25, wherein notifying the user comprises halting execution of the code.

29. (Previously Presented) The computer readable storage medium of claim 25, wherein the upper limit is independent of other memory size limitations.

30. (Previously Presented) The computer readable storage medium of claim 25, wherein the upper limit is not a limit on a stack size.

31. (Currently Amended) A computer system comprising: an output device, a memory device, one or more processors, code resident in the memory device and containing a plurality of memory allocator calls and a plurality of memory deallocator calls, a heap manager resident in the memory device to allocate and free memory of the memory device, ~~and~~ a debugger program resident in the memory device~~;~~, and a data structure resident in the memory device and configured to record relationships between the memory deallocator calls and the memory allocator calls, wherein each of the relationships specify a memory allocator call and a memory deallocator call, and wherein each of the relationships requires at least one of: (i) that memory space allocated by the specified memory allocator call is freed by the specified memory deallocator call; and (ii) that memory space freed by the specified memory deallocator call was allocated by the specified memory allocator call; wherein the data structure stores, for each relationship, a reference to both the specified memory deallocator call and the specified memory allocator call of the respective relationship, and wherein the debugger program ~~comprising~~ comprises a debugger user interface configured to at least:

allow a user to view allocation/deallocation history information at a user-specified memory location; and

allow a user to establish a relationship between a memory deallocator call and a memory allocator call, wherein the relationship is stored in the data structure, ~~wherein the relationship is represented by a data structure containing a reference to the user-~~

~~selected deallocator and the user-selected allocator, and wherein the relationship requires that memory space allocated by the memory allocator call is freed by the memory deallocator call~~ and wherein a violation of ~~the~~ a requirement of the relationship causes the debugger user interface to notify the user.